08/437975

Attorney's Docket No.: __018414-082__

APPLICATION FOR UNITED STATES LETTERS PATENT

BY

WILLIAM K. ZURAVLEFF

and

TIMOTHY ROBINSON

for

A CONTROLLER FOR A SYNCHRONOUS DRAM

A CONTROLLER FOR A SYNCHRONOUS DRAM

BACKGROUND

The present invention is directed to a controller for maximizing throughput of memory requests from an external device to a synchronous

5 DRAM. More particularly, the present invention is directed to a controller which prioritizes multiple memory requests from the external device and issues reordered memory requests to the synchronous DRAM so that the throughput from the external device to the synchronous DRAM is maximized.

Synchronous DRAMs are relatively new devices which are similar to

10 conventional DRAMs but the synchronous DRAMs have some important differences. The architecture of the synchronous DRAMs is similar to conventional DRAMs. For instance, the synchronous DRAMs have multiplexed address pins, control pins such as RAS, CAS, CS, WE, and bidirectional data pins. Also, the synchronous DRAMs activate a page as does the conventional

15 DRAM and then subsequent accesses to that page occur faster. Accordingly, a precharge operation must be performed before another page is activated.

One difference between synchronous DRAMs and conventional DRAMs is that all input signals are required to have a set-up and hold time with respect to the clock input in synchronous DRAMs.   The hold time is referenced to the

20 same clock input. The outputs also have a clock to output delay referenced to the same clock. Thereby, the synchronous characteristics are provided. Furthermore, the synchronous DRAMs are pipelined which means that the latency is generally greater than one clock cycle. As a result, second and third synchronous DRAM commands can be sent before the data from the original

25 write request arrives at the synchronous DRAM. Also, the synchronous DRAMs have two internal banks of data paths which generally correspond to separate memory arrays sharing I/O pins. The two internal banks of memory paths are a JEDEC standard for synchronous DRAMs. An example of a known synchronous

DRAM is a 2 MEG x 8 SDRAM from Micron Semiconductor, Inc., model no. MT48LC2M8SS1S.

In the synchronous DRAMs, almost all I/O timings are referenced to the input clock. Minimum parameters such as CAS latency remain but are

5   transformed from electrical timing requirements to logical requirements so that they are an integral number of clock cycles. The synchronous DRAMs for at least by-four and by-eight parts are a JEDEC standard with defined pin outs and logical functions. Because the synchronous DRAMs are internally pipelined, the pipe stage time is less than the minimum latency so that spare time slots can be

10   used for other functions. For instance, the spare time slots can be used for bursting out more data (similar to a nibble mode) and issuing another "command" with limitations.

Certain problems arise when using synchronous DRAMs which must be addressed. For instance, the clock to output delay can equal the whole cycle.

15   Also, because the synchronous DRAMs are pipelined, a second request must be given before the first one is complete to achieve full performance. Furthermore, the output electrical/load/timing specifications of synchronous DRAMs are difficult to meet. Therefore, a controller is desired for interfacing the synchronous DRAMs with devices which read and write, such as

20   microprocessors, and meeting JEDEC standards for synchronous DRAMs so that versatile synchronous DRAMs may be provided and applied in many design applications.

<div align="center">SUMMARY</div>

An object of the present invention is to control a synchronous DRAM by

25   interfacing an external device, such as a microprocessor, for reading and writing to the synchronous DRAM.

Another object of the present invention is to provide a controller for a synchronous DRAM which can buffer and process multiple memory requests so

3

that greater throughput, or an equivalent throughput at less latency, can be achieved by the synchronous DRAM.

A still further object of the present invention is to provide a controller for issuing and completing requests out of order with respect to the received or

5   issued order so that the throughput of the synchronous DRAM is improved by overlapping required operations, which do not specifically involve data transfer, with operations involving data transfer.

A still further object of the present invention is to provide a controller for a synchronous DRAM that schedules memory request commands as closely

10   together as possible within the timing constraints of the synchronous DRAM so that the throughput of the memory requests is maximized.

These objects of the present invention are fulfilled by providing a controller for a synchronous DRAM comprising a sorting unit for receiving memory requests and sorting.said memory requests based on their addresses and

15   a throughput maximizing unit for processing said memory requests to the synchronous DRAM in response to scheduling which maximizes the use of data slots by the synchronous DRAM. The controller is able to prioritize and issue multiple requests to the synchronous DRAM in a different order than was received or issued such that the out of order memory requests improve the

20   throughput to the synchronous DRAM. In particular, the controller issues memory requests as closely together as possible while maintaining the timing constraints of the synchronous DRAM based on its specifications.

The objects of the present invention are also fulfilled by providing a method for controlling a synchronous DRAM comprising the steps of receiving

25   memory requests and sorting said memory requests based on their addresses, and maximizing throughput of said memory requests to the synchronous DRAM so that use of data slots by the synchronous DRAM is maximized. Similarly, this method controls the memory requests issued to the synchronous DRAM so that

/_/

they are spaced as closely as possible while maintaining the timing constraints of the synchronous DRAM based on its specifications.

Further scope of applicability of the present invention will become apparent from the detailed description given hereinafter. However, it should be

5   understood that the detailed description and specific examples, while indicating preferred embodiments of the invention, are given by way of illustration only, since various changes and modifications within the spirit and scope of the invention will become apparent to those skilled in the art from this detailed description.

10   **BRIEF DESCRIPTION OF THE DRAWINGS**

The present invention will become more fully understood from the detailed description given hereinbelow and the accompanying drawings which are given by way of illustration only, and thus are not limitative of the present invention, wherein:

15   Figure 1 illustrates a controller for a synchronous DRAM according to an embodiment of the present invention;

Figure 2 illustrates a block diagram for prioritizing and issuing multiple requests to the synchronous DRAM by the controller illustrated for an embodiment of the present invention;

20   Figures 3(a) - 3(h) illustrate part configurations of the synchronous DRAM for embodiments of the present invention; and

Figures 4(a) - 4(c) illustrate timing diagrams for commands to the synchronous DRAM.

                    DETAILED DESCRIPTION

25   Figure 1 illustrates a controller for a synchronous DRAM according to an embodiment of the present invention. In Figure 1, signals corresponding to memory requests and commands from an external device, such as a microprocessor for example, are input to the controller at a bank sort unit 10. The bank sort unit 10 processes address, control, and data signals and sorts the

data based on the address before sending the data to bank data paths 20 and 30. The output of the bank data paths 20 and 30 are multiplexed by a multiplexer 40 and input to a pad driver 50 before being input to the synchronous DRAM 100. The memory requests and commands are tagged to indicate their received order.

5   The tags are used to indicate the order of the memory requests and commands for achieving increased throughput or re-ordering the memory requests and commands as will be further described. The tags may be associated with the memory requests and commands at the external device or may be generated by the bank sort unit 10. A return data path 60 receives data from the bank data

10   paths 20 and 30 and the synchronous DRAM 100 via the pad driver 50 and is used to process and re-order the data, if necessary, based on the tag information.

The controller also includes a control block 70 which receives a controller clock. The control block 70 outputs a slower SDRAM clock which is based on the controller clock. More specifically, the SDRAM clock is derived in the

15   control block 70 by dividing the controller clock with a programmable divisor between 4 and 32, inclusive, and applying wave shaping circuits having programmable offset and duty cycle inputs. The SDRAM clock is then input to the synchronous DRAM 100.

The controller may be implemented in BiCMOS technology which

20   includes bipolar (ECL) style logic with CMOS pad drivers having integrated low swing differential to full swing (LVTTL) voltage level converters in a preferred embodiment. Because the logic is implemented in the bipolar (ECL) style circuits, very high speeds are possible with cycle times being in excess of 1 GHz. The controller is designed to run at a clock frequency greater than the

25   SDRAM clock frequency. Both the SDRAM clock and sampling point for incoming data can be controlled to the accuracy of the controller clock so that a fully synchronous circuit can be maintained which has a sample point resolution greater than the SDRAM clock.

The programmable sample point overcomes the problem of the SDRAM clock to output delay being greater than or equal to one SDRAM clock period by providing a sampling point which can be placed at a particular controller clock edge. In the present controller, no analog delay elements are required or used.

5   If analog delay elements were to be used in a different controller, which has a single clock frequency for the controller and the synchronous DRAM for example, the data generally would have to be resynchronized to the single clock after sampling. As a result, the data would be delayed by another entire SDRAM clock period before the data is available for use by the processor if

10   analog delay elements are used. In addition, the return data path 60 runs at the same timing as the SDRAM clock but can be offset in time from the SDRAM clock edges by a programmable number of controller clock edges. The programmable clock and sampling points can also be used to provide adequate timing margins for different printed circuit board trace lengths or different

15   loading conditions from various part configurations.

At the input from the external device, or within the controller, each memory request is "tagged" or assigned an integer to indicate the order in which the memory request was received as part of the control stream. This tag has three purposes. One purpose is for the tag to be passed with the loaded data

20   back to the external device so that loads which are returned out of order will be indicated. A second purpose is that the tag may be used to send the earliest request to the synchronous DRAM when multiple pending requests may be serviced. A third purpose is that the tag may be used to service the earliest pending request only if the return of load data is required to be in order such as

25   during system integration or debug. The controller functions to translate memory load and store requests into synchronous DRAM commands. For example, a simple load might be translated to the synchronous DRAM as the following sequence of commands: precharge, activate (for the high portion of addresses), and read (for the low portion of addresses). The controller also

functions to enforce timing requirements and to queue a minimum of two requests. Furthermore, the controller functions to interrupt normal operation for required refresh, power on reset (load mode register), power down, and power up. The main purpose of the controller is to allow or provide greater

5   throughput, or equivalent throughput at less latency, from the synchronous DRAMs.

This controller is applicable to computer systems where processors can request data much faster than the memory can provide the data (generally four to fifty times faster). Also, the controller is applicable to processors which do not

10   stall for an individual memory request or processors which do not stop to wait for data where possible. The main goal of the controller is to achieve a high throughput of the memory requests and commands. Another important issue is latency, but minimizing latency is secondary to maximizing throughput in the present controller. Another goal of the controller is to provide flexibility in

15   timing and configuration, but this is also secondary to maximizing throughput.

One feature of the controller is to buffer multiple memory requests, and in an example of the present embodiment, two memory requests can be buffered per bank. Buffering up to one request per bank allows for the opportunity of the banks being in parallel. By buffering two requests to the same bank, the same

20   page address match computation and the actual memory data transfer may be placed in parallel. Another feature of the controller is to issue and complete requests out of order with respect to the order received or the order issued by the external device. Out of order issue/completion can improve throughput by overlapping required operations which do not specifically involve data transfer

25   with operations involving data transfer.

The controller for state machines implemented in digital logic allows for out of order issue and completion by using dynamic constraint based, maximum throughput scheduling. In general, the memory requests are resolved into their required sequences of precharge/bank activate/read-write sequences and placed in

a queue where one queue per bank exists. Dynamic constraint based logic

determines which requests are ready to issue and scheduling logic chooses the

requests thereafter, but not necessarily in the order that the requests are received.

The requests are chosen according to the requests which can be issued to the

5   synchronous DRAM without causing reduction in throughput. For example,

bank 1 may be precharged while bank 0 is bursting out data. Also, the

scheduling and logic select requests such that the throughput is maximized. For

example, a read to a page and bank is chosen when directly following a read to

that same page and bank.

10   Figure 2 further illustrates how multiple requests are prioritized and

issued to the synchronous DRAMs. Each of the units illustrated in Figure 2 are

incorporated into the controller. For example, a command update unit 210, a

bank qualification unit 220 and a constraint update unit 240 may be incorporated

into each of the bank data paths 20 and 30 of Figure 1 and a bank arbitration unit

15   230 may be incorporated into the control block 70. There is one bank arbitration

unit 230 and a plurality (N) of other command update, bank qualification and

constraint update units 210, 220 and 240 where N=the number of banks. The

bank qualification unit 220 qualifies the per bank requests with the current

synchronous DRAM status (active or precharged) and ongoing scheduling

20   constraints. The bank qualification unit 220 further interprets memory requests

and commands, such as load and store, into synchronous DRAM commands,

such as activate, read, write and precharge. The bank arbitration unit 230 is

connected to the bank qualification unit 220 and arbitrates between multiple

sources of memory requests. The constraint update unit 240 is connected to the

25   bank arbitration unit 230 for decoding the decision to synchronous DRAM

standard commands while simultaneously updating scheduling constraints per

each separate bank. The command update unit 210 is connected to the constraint

update unit 240 for updating the per bank memory requests queues and popping

the top element off to reveal the next request when necessary.

The functions for each of the units described above do not have to be accomplished within four pulses of the controller clock, or a certain number of pulses of the controller clock. However, these steps must be completed within one SDRAM clock pulse. The SDRAM clock pulse relates to the controller clock and is determined by dividing the controller clock with the programmable divisor. A command stack of memory requests is developed for each bank of the data paths. The command update unit 210 updates the per bank command stack. An illustration of the update performed is provided in Table 1 below for a two-deep control stack.

TABLE 1
COMMAND UPDATE

Incoming Request						TOS		Next				TOS		Next	
go	rw	np	pop	pnp	st	tr	tnp	nr	nnp	st	tr	tnp	nr	nnp
incoming transactions, no pop's														
1	X	X	0	0	0	X	X	X	X	1	rw	np	X	X
1	X	X	0	0	1	X	X	X	X	2	tr	tnp	rw	np
pop's														
0	X	X	1	0	1	X	X	X	X	0	X	X	X	X
0	X	X	1	0	2	X	X	X	X	1	nr	nnp	X	X
0	X	X	0	1	1	X	X	X	X	1	tr	0	X	X
0	X	X	0	1	2	X	X	X	X	2	tr	0	nr	nnp
simultaneous incoming transactions and pops														
1	X	X	1	0	1	X	X	X	X	1	rw	np	X	X
1	X	X	0	1	1	X	X	X	X	2	tr	0	rw	np
do nothing														
0	X	X	0	0	0	X	X	X	X	0	X	X	X	X
0	X	X	0	0	1	X	X	X	X	1	tr	tnp	X	X
0	X	X	0	0	2	X	X	X	X	2	tr	tnp	nr	nnp

In Table 1, go represents the existence of a new command from the external device, rw represents a read or write request where rw = 1 indicates a read request and rw = 0 indicates a write request, np represents a new page

where the transaction is a new page (row) address, pop represents an input transaction for popping the stack, pnp represents an input transaction for clearing the np bit if set (this allows a page activate to clear the np bit without popping the stack), and st represents the number of elements in the stack. Also, TOS

5   indicates top of stack and next indicates the next entries in the stack.

In all of the tables, the inputs are on the left of the vertical bar and the outputs are to the right. An input of "X" means "don't care" and an output of "X" means unspecified. Outputs may assume symbolic values representing specific numeric values and may also assume the value of an input where the

10   name of the input appears in the table.

The bank qualification unit 220 qualifies the top of the stack with constraints of the synchronous DRAM. The constraints on the synchronous DRAM may be obtained, for example, from the specification sheet of the device used, such as from the Micron Semiconductor Specification Sheet for

15   MT48LC2M8S1S. The memory requests and commands are qualified per each bank of data path. Bank control inputs (st), read (tr) and new page (tnp), are combined with bank active signals and qualifiers (ok to read, write, activate, and precharge) to develop a per bank command request as illustrated in Table 2 below. The first three inputs of Table 2 correspond to the first three outputs in

20   Table 1 from the command update unit 210. The control inputs indicate that there is a request pending (st > 0), a read request is pending when the read qualifier is active and a new page address is requested when the new page qualifier is active.

TABLE 2
BANK QUALIFICATION

st	tr	tnp	TOS Active	OK to Read Bank 0	OK to Write Bank 0	OK to Activate Bank 0	OK to Prechg Bank 0	Precharge if Idle	B State
new page									
>0	X	1	0	X	X	1	X	X	BA1

25

Docket No.: 018414-082

st	TOS		Active	OK to Read Bank 0	OK to Write Bank 0	OK to Activate Bank 0	OK to Prechg Bank 0	Precharge if Idle	B State
	tr	tnp							
>0	X	1	0	X	X	0	X	X	BI1
>0	X	1	1	X	X	X	1	X	BP1
>0	X	1	1	X	X	X	0	X	BI1

write

>0	0	0	0	X	X	1	X	X	BA1
>0	0	0	1	X	1	X	X	X	BW1
>0	0	0	0	X	X	0	X	X	BI1
>0	0	0	1	X	0	X	X	X	BI1

read

>0	1	0	0	X	X	1	X	X	BA1
>0	1	0	1	1	X	X	X	X	BR1
>0	1	0	0	X	X	0	X	X	BI1
>0	1	0	1	0	X	X	X	X	BI1

if no command, precharge if active, otherwise idle

0	X	X	0	X	X	X	X	X	BI1
0	X	X	1	X	X	X	1	1	BP1
0	X	X	1	X	X	X	1	0	BI1
0	X	X	1	X	X	X	0	X	BI1

In Table 2, OK to Read Bank 0 is a constraint for reading from bank 0 which indicates that it is possible to read in this SDRAM cycle, Write Bank 0 is a constraint for writing to bank 0 which indicates that it is possible to write in this SDRAM cycle, OK to Activate Bank 0 is a constraint for activating bank 0 which indicates that it is possible to activate in this SDRAM cycle and Prechg Bank 0 is a constraint for precharging bank 0 which indicates that it is possible to precharge in this SDRAM cycle.

Next, the bank arbitration unit 230 arbitrates between commands from different banks so that the throughput is maximized. The arbitration is performed after the qualification so that the data slots between the commands are

Docket No.: 018414-082

used as much as possible for memory requests. The arbitration is performed between banks and between system conditions such as power up, power down, and refresh. Table 3 below illustrates that one of the conflicting memory requests or commands is selected to maintain throughput based on previous reads or writes (with read always being picked over write) for preventing unused data slots. The first two inputs of Table 3 correspond to the outputs from the bank qualification unit 220 in Table 2. In addition, a least recently used bit (lrub) of the current bank is used as a "tie-breaker" for choosing a request.

TABLE 3
BANK ARBITRATION

B State Bank 0	B State Bank 1	Supervisory State	Current Bank	Current Bank	C-State
BW1	BR1	SI	X	1	R1
BA1	BR1	SI	X	1	R1
BR1	BW1	SI	X	0	R1
BR1	BA1	SI	X	0	R1
BR1	BP1	SI	X	0	R1
BA1	BW1	SI	X	1	W1
BP1	BW1	SI	X	1	W1
BW1	BA1	SI	X	0	W1
BW1	BP1	SI	X	0	W1
BP1	BA1	SI	X	1	A1
BA1	BP1	SI	X	0	A1

both banks want to do the same thing so preference is given to the least recently used bank

BR1	BR1	SI	0	1	R1
BR1	BR1	SI	1	0	R1
BW1	BW1	SI	0	1	W1
BW1	BW1	SI	1	0	W1
BA1	BA1	SI	0	1	A1
BA1	BA1	SI	1	0	A1
BP1	BP1	SI	0	1	P1
BP1	BP1	SI	1	0	P1

arbitrate control state when one or more banks request idle

| BI1 | BI1 | SI | X | cb | I1 |
| BR1 | BI1 | SI | X | 1 | R1 |

B State Bank 0	B State Bank 1	Supervisory State	Current Bank	Current Bank	C-State
BW1	BI1	SI	X	0	W1
BA1	BI1	SI	X	0	A1
BP1	BI1	SI	X	0	P1
BI1	BR1	SI	X	1	R1
BI1	BW1	SI	X	1	W1
BI1	BA1	SI	X	1	A1
BI1	BP1	SI	X	1	P1

reset sequence reqeusts, if active, have precedence

X	X	SP	X	cb	P1
X	X	SRF	X	cb	RF1
X	X	SPD	X	cb	PD1
X	X	SM	X	cb	M1
X	X	SW	X	cb	I1
X	X	SIL	X	cb	I1

In Table 3, B-State Bank 0 represents the bank state of bank 0 and B-state Bank 1 represents the bank state of bank 1. These two inputs along with the supervisory state have three bit wide field widths. The current bank select bit is at the input and the next bank select bit at the output and C-state represents the arbitrated control state.

The constraint update unit 240 is used to update constraints. Table 4 below illustrates how constraints are updated in each bank in response to the C-State output from the bank arbitration. The output counter indicates the number of SDRAM cycles one must wait before a read, write, activate or precharge operation can be performed.

TABLE 4
CONSTRAINT UPDATE

BL (Burst Length)	t RCD (R/C Delay)	C State	Current Bank	Counter	OK to Read
X	2	A1	1	trCD-1	0
X	3	A1	1	tRCD-1	0
X	X	R1	X	BL-1	0

BL (Burst Length)	t RCD (R/C Delay)	C State	Current Bank	Counter	OK to Read
X	X	W1	1	BL-1	0

tAA (CAS Latency)	BL	t RCD	C-State	Current Bank	Counter	OK to Write
X	X	2	A1	1	tRCD-1	0
X	X	3	A1	1	tRCD-1	0
X	X	X	R1	X	tAA + BL-1	0
X	X	X	W1	X	BL-1	0

tRCm1 (Read Cycle Time)	tRRD (Row-Row Delay)	tRP (RAS Precharge)	C-State	Current Bank	Counter	OK to Activate
X	X	X	M1	X	tRCm1	0
X	X	X	RF1	X	tRCm1	0
X	X	0..1	P1	1	0	1
X	X	>1	P1	1	tRP-1	0
X	0..1	X	A1	0	0	1
X	>1	X	A1	0	tRRD-1	0

tAA	BL	tWR (Write Recovery Time)	C-State	Current Bank	Counter	OK to Precharge
0..1	<4	X	R1	1	0	1
0..1	4	X	R1	1	tAA + 2	0
>1	X	X	R1	1	tAA + BL-3	0
X	X	X	W1	1	tWR + BL-2	0

/5

Figures 3(a) - 3(h) illustrate some examples of part configurations supported by the controller. The part configurations of Figures 3(a), 3(c), 3(e) and 3(g) use 16 Mbit parts to support 4MBytes, 2MBytes, 8MBytes and 4MBytes of total memory respectively. The part configurations of Figures 3(b), 3(d), 3(f)

5   and 3(h) use 64 Mbit parts to support 16MBytes, 8MBytes, 32MBytes and 16MBytes of total memory respectively. According to JEDEC specifications, the synchronous DRAM has two banks. As a result, the maximum number of banks is two times the number of parts used. With more banks, a more random stream of data can be handled faster. However, as the number of banks used increases,

10   the hardware complexity increases due to the larger number of decisions which must be made at the same time.

In Figures 3(a) - 3(h), a microprocessor 300 is connected to synchronous DRAMs for the various configurations. The configurations of Figure 3(a) and 3(b) support two 16Mbit synchronous DRAMs 310 and 312 and two 64Mbit

15   synchronous DRAMs 314 and 316, respectively. The configurations of Figures 3(c) and 3(d) support one 16Mbit synchronous DRAMs 320 and one 64Mbit synchronous DRAMs 322, respectively. In Figures 3(e) and 3(f), configurations of four synchronous DRAMs 330, 332, 334 and 336 and four synchronous DRAMs 340, 342, 344 and 346 are respectively supported. Figures 3(g) and

20   3(h) support configurations of two synchronous DRAMs 350 and 352 and two synchronous DRAMs 354 and 356, respectively.

Figure 4(a) illustrates the timing signals to the synchronous DRAM over a relatively large time frame. The controller clock (the fast clock) and the slower SDRAM clock are illustrated for the power up period and subsequent stores and

25   loads in addition to signal at the SDRAM pins.

Figure 4(b) illustrates an example of the reordering of a store request that occurs in the controller before being issued in the synchronous DRAM. As illustrated by the controller input bus, three writes are to be performed to bank 0 and one write is to be performed to bank 1. Also illustrated are the scheduled no

16

operation periods (NOPs) which correspond to the timing constraints of the synchronous DRAM. For the first request on a controller input bus, bank 0 is activated and then written into. Next, a write operation is performed to bank 0 and no activation is necessary since bank 0 has already been activated. A write

5 request follows to bank 1 and bank 1 must be activated as a result. Due to the timing constraints, bank 1 must wait to be written into. However, bank 0 still may be written into and since the request immediately following the write request to bank 1 is a write request to bank 0, the write request to bank 0 may be performed immediately after bank 1 is activated. Thereafter, the write to bank 1

10 may be performed after waiting the required amount of time after the activation to bank 1. Accordingly, the data slots are used as much as possible to maximize the throughput to the synchronous DRAM for the store operation.

Figure 4(c) similarly illustrates a timing diagram for loads. The first three requests are to bank 0 and after activating bank 0, the three read operations

15 are successively performed. After the third load request to bank 0, a load request to bank 1 is requested. Therefore, bank 1 must be activated and the required amount of time must follow this activation. However, immediately after the load request to bank 1, a load request to bank 0 is requested. Accordingly, the load request to bank 0 may be immediately performed since the required time

20 following the bank 1 activation must be met. After completing the read to bank 0 and waiting the sufficient amount of time for completing the activation, the read is performed to bank 1. Thereby, throughput of the memory requests to the synchronous DRAM is maximized. In contrast to stores, the return load data should be associated with the correct load request. Therefore, the tagging of the

25 memory requests is important to ensure that returning load data is associated with the correct load request.

The invention being thus described, it would be obvious that the same may be varied in many ways. Such variations are not to be regarded as a departure from the spirit and scope of the invention and all such modifications

17

that would be obvious to one skilled in the art are intended to be included within the scope of the following claims.

18